Amendments to the Claims:

The following listing of claims will replace all prior versions and listings of claims in the application:

1. (Currently Amended) A method comprising:

providing asynchronous access to multiple users to a graphical programming and analysis environment program visually represented as a white board;

~~allowing~~ wherein each user of the multiple users ~~to generate~~ generates graphically represented code objects within the environment program, to further ~~comprising~~ comprise:

~~allowing~~ said each user ~~to instantiate~~ instantiates said one or more code objects,

~~allowing~~ said each user ~~to determine~~ determines an internal logic for each code object of said one or more code objects,

~~allowing~~ said each user ~~to determine~~ determines first data to be received by said each code object, and

~~allowing~~ said each user ~~to determine~~ determines second data to be sent by said each code object;

~~allowing~~ wherein said each user obtains access to the code objects of other users of the multiple users based on security privileges accorded to said each user, in which the code objects reported over a network are hierarchically filtered based on the security privileges according to said each user, and the security privileges restrict those functions or features of the code objects available to said each user;

~~allowing~~ wherein said each user ~~to have~~ chains the code objects of said each user ~~be chained~~ to the code objects of the other users to which said each user has access to yield inter-code object communication by inter-code object connections, each inter-code object connection terminating on one of an edge and an interior of one of the code objects; ~~and,~~ and

~~allowing~~ wherein said each user ~~to~~ executes an ~~execute~~ application program composed of the code objects as chained together within the environment program, the program operating by:

opening a first input window that displays a first dialog box and a first acknowledgement cursor region, wherein the first dialog box receives the first data, and the internal logic receives the first data in response to said each user executing the first acknowledgement cursor region,

opening a second input window that displays a second dialog box and a second acknowledgement cursor region, wherein the second dialog box receives the second data, and the internal logic receives the second data in response to said each user executing the second acknowledgement cursor region, and

opening an output window that displays result data from the internal logic operating on the first and second data, wherein the result data reported over the network are hierarchically filtered based on the security privileges;

opening a chat area within which said each user can communicate with the other users; and

opening a user list area that displays a name of the multiple users that are currently logged into the environment program.

2. (Original) The method of claim 1, wherein providing asynchronous access to the multiple users to the graphical programming and analysis environment program comprises enabling multiple users to log into the environment program remotely, such that the multiple users are able to access the environment program simultaneously.

3. (Cancelled)

4. (Previously Presented) The method of claim 1, wherein allowing said each user access to the code objects of the other users based on security privileges accorded to the user comprises rendering visible to said each user the code objects of the other users to which the user has access.

5. (Previously Presented) The method of claim 1, wherein allowing said each user to chain together the code objects of the user to the code objects of the other users to which the user has access comprises allowing the user to graphically chain code objects together, such that

a sender object of a pair of graphically chained together code objects is able to send data that are received by a receiver object of the pair.

6. (Previously Presented) The method of claim 1, wherein allowing said each user to execute the application program composed of the code objects as chained together within the environment program comprises displaying to the user end results of data processed by the code objects upon execution of the application programs.

7. (Previously Presented) The method of claim 1, wherein the graphical programming and analysis environment program comprises an applet program, and said each code object comprises an applet program within a same applet context as the environment program.

8. (Original) The method of claim 7, wherein at least one of the graphical programming and analysis environment program and the code objects is developed within an architecture-independent and Internet web browsing program-independent computer programming technology.

9. (Original) The method of claim 1, wherein the graphically represented code objects coexist with non-graphically represented code objects within the environment program.

10. (Original) The method of claim 9, wherein the non-graphically represented code objects comprise stand-alone computer programs.

11. (Original) The method of claim 9, wherein the non-graphically represented code objects comprise one or more of: image-viewing programs, video-playing programs, and audio-playing programs.

12. (Original) The method of claim 1, wherein the graphically represented code objects comprise one or more of: database objects, video-playing programs, audio-playing programs, image-viewing programs, geo-spatial information map-viewing programs, filter-algorithm programs, and model and analysis tool programs.

13. (Cancelled)

14. (Previously Presented) The method of claim 1, wherein providing asynchronous access to the graphical programming and analysis environment program comprises providing application program executable within the white board.

15. (Previously Presented) The method of claim 14, wherein providing the application programs executable within the white board comprises executing the application program such that results thereof are immediately available to the multiple users.

16. (Original) The method of claim 1, wherein providing asynchronous access to the graphical programming and analysis environment program comprises allowing users to access resources available on a network to which the graphical programming and analysis environment program is communicatively coupled.

17. (Currently Amended) An apparatus to provide an environment for multiple-user graphical programming and analysis by machine-readable instructions executable on a computer platform, said apparatus comprising:

at least one processor for creating a plurality of graphically represented code objects, each code object created by a user and accessible by other users in accordance with security privileges of the other users, said each code object comprises:

a data interface indicating first data to be input into the code object and second data to be output by the code object, and

internal logic to generate the second data from the first data;

said processor for creating a plurality of graphically represented inter-code object connections, each inter-code object connection representing data transfer between a pair of code objects;

said processor for creating at least one application program composed of one or more chains of the code objects interconnected via the inter-code object connections, the program including opening operations for:

a first input window for displaying a first dialog box and a first acknow-

ledgement cursor region, wherein the first dialog box receives the first data, and the internal logic receives the first data in response to said each user executing the first acknowledgement cursor region,

a second input window for displaying a second dialog box and a second acknowledgement cursor region, wherein the second dialog box receives the second data, and the internal logic receives the second data in response to said each user executing the second acknowledgement cursor region, and

an output window for displaying result data from the internal logic operating on the first and second data; ~~and,~~

said processor for creating a graphical white board area within which to dispose the code objects ~~are definable and movable~~ and to create the inter-code object connections ~~are creatable~~;

said processor for creating a chat area within which the user can communicate with the other users; and

said processor for creating a user list area showing a name of each of the user and the other users currently logged into the environment program,

wherein the application program is executable within the graphical white board area, and each inter-code object connection terminates on one of an edge and an interior of one of the code objects, and

wherein said each user obtains access to the code objects of other users based on security privileges accorded to the user, in which the code objects reported over a network are hierarchically filtered based on the security privileges according to the user, and the security privileges restrict those functions or features of the code objects available to the user.

18. (Previously Presented) The apparatus of claim 17, wherein each code object is an applet program.

19. (Previously Presented) The apparatus of claim 18, wherein the graphical white board area is an applet program having a context within which each code object runs.

20. (Previously Presented) The apparatus of claim 18, wherein the applet program is a JAVA® applet program.

21. (Cancelled)

22. (Previously Presented) The apparatus of claim 17, wherein each code object has at least one inter-code object communication graphically terminating on one of an edge and an interior of the code object.

23. (Previously Presented) The apparatus of claim 17, wherein each inter-code object connection represents data being sent by a sender object of the pair of code objects and being received by a receiver object of the pair of code objects.

24. (Previously Presented) The apparatus of claim 23, wherein the data are at least one of: user defined, and filtered according to security privileges accorded to the users.

25. (Cancelled)

26. (Previously Presented) The apparatus of claim 17, wherein at least one of the inter-code object connections is one of graphically invisible and purposefully limited in functionality for security.

27. (Previously Presented) The apparatus of claim 17, wherein each inter-code object connection is graphically represented by one of a line and a directed graph.

28. (Previously Presented) The apparatus of claim 17, wherein the at least one application program is constructed one of asynchronously and synchronously.

29. (Previously Presented) The apparatus of claim 17, wherein the at least one application program is at least one of: capable of being stored for later retrieval and use, and modular in nature so that more complex application programs may be constructed therefrom.

30. (Previously Presented) The apparatus of claim 17, wherein the at least one application program is contained within container panels as macro programs, the container panels interconnectable via additional inter-code object connections.

31. (Previously Presented) The apparatus of claim 17, wherein the at least one application program is at least one of: auditable and loggable during usage, traceable to users who construct the programs, traceable to users who use the programs, and configuration manageable.

32. (Previously Presented) The apparatus of claim 17, wherein the at least one application program is at least one of:

capable of accepting data from dynamically changing input sources, from static input sources, and from network-accessible resources;

capable of network reporting results thereof; and,

capable of networking reporting security privilege-filtered results thereof.

33. (Cancelled)

34. (Currently Amended) A computer-implemented method comprising:

accessing by a user a graphical programming and analysis environment program that other users are already currently accessing;

generating by the user graphically represented code objects within the environment program, wherein for each code object,

the user determining a data interface indicating first data to be input into the code object and second data to be output by the code object; ~~and,~~ and

the user determining internal logic to generate the second data from the first data;

graphically chaining together the code objects by the user within the environment program, including chaining together the code objects generated by the user and code objects generated by the other users to which the user has access based on security privileges accorded to the user, to yield inter-code object communication by inter-code object connections, each inter-

code object connection terminating on one of an edge and an interior of one of the code objects; ~~and,~~

hierarchically filtering the code objects reported over a network based on the security privileges according to the user; and

assembling an application program by the user within the environment program, each application program composed of the code objects ~~as have been~~ that are chained together, the application program operating by:

opening a first input window that displays a first dialog box and a first acknowledgement cursor region, wherein the first dialog box receives the first data, and the internal logic receives the first data in response to said each user executing the first acknowledgement cursor region,

opening a second input window that displays a second dialog box and a second acknowledgement cursor region, wherein the second dialog box receives the second data, and the internal logic receives the second data in response to said each user executing the second acknowledgement cursor region, ~~and~~

opening an output window that displays result data from the internal logic operating on the first and second data,

opening a chat area within which said each user can communicate with the other users;

opening a user list area that displays a name of the multiple users that are currently logged into the environment program, and

providing access for said each user to the code objects of the other users of the multiple users based on security privileges accorded to the user, wherein the security privileges restrict those functions or features of the code objects available to the user.

35. (Original) The method of claim 34, further comprising executing by the user of the application programs within the environment program.

36. (Cancelled)

37. (Original) The method of claim 34, wherein chaining together code objects by the user comprises the user, for each pair of code objects to be chained together, specify a sender object of the pair to send data and a receiver object of the pair to receive the data.

38. (Original) The method of claim 34, wherein the graphical programming and analysis environment program comprises an applet program, and each code object comprises an applet program within a same applet context as the environment program.

39. (Previously Presented) The method of claim 1, wherein opening an output window further includes displaying a third acknowledgement cursor region thereby allowing said each user to terminate the output window in response to said each user executing the third acknowledgement cursor region.

40. (Previously Presented) The apparatus of claim 17, wherein the output window further includes a third acknowledgement cursor region that allows said each user to terminate the output window in response to said each user executing the third acknowledgement cursor region.

41. (Previously Presented) The method of claim 34, wherein opening an output window further includes displaying a third acknowledgement cursor region thereby allowing said each user to terminate the output window in response to said each user executing the third acknowledgement cursor region.

42. (New) The apparatus of claim 17, wherein an omnibus processor combines operations of said at least one processor.